Marked-up copy of As filed Application

SYSTEM AND METHOD OF PARALLEL LOADFLOW COMPUTATION FOR ELECTRICAL POWER SYSTEM

TECHNICAL FIELD

- paragraph numbers are added

[001] The present invention relates to methods of loadflow computation in power flow control and voltage control in an electrical power system. It also relates to the parallel computer architecture and distributed computing architecture.

BACKGROUND ~~ART AND MOTIVATION~~ & OF THE ENVENTION

Utility/industrial power networks are composed of many power plants/generators interconnected through transmission/distribution lines to other loads and motors. Each of these components of the power network is protected against unhealthy or alternatively faulty, over/under voltage, and/or over loaded damaging operating conditions. Such a protection is automatic and operates without the consent of power network operator, and takes an unhealthy component out of service disconnecting it from the network. The time domain of operation of the protection is of the order of milliseconds.

[002] The purpose of a utility/industrial power network is to meet the electricity demands of its various consumers 24-hours a day, 7-days a week while maintaining the quality of electricity supply. The quality of electricity supply means the consumer demands is met at specified voltage and frequency levels without over loaded, under/over voltage operation of any of the power network components. The operation of a power network is different at different times due to changing consumer demands and/or development of any faulty/contingency situation. In other words healthy operating power network is constantly subjected to small and large disturbances. These disturbances could be consumer/operator initiated, or initiated by overload and/or under/over voltage alleviating functions collectively referred to as security control functions and various optimization functions such as economic operation and minimization of losses, or caused by a fault/contingency incident.

[003] For example, a power network is operating healthy and meeting quality electricity needs of its consumers. A fault occurs on a line or a transformer or a generator which faulty component gets isolated from the rest of the healthy network by virtue of the automatic

operation of its protection. Such a disturbance would cause a change in the pattern of power flows in the network, which can cause over loading of one or more of the other components and/or over/under voltage at one or more nodes in the rest of the network. This in turn can isolate one or more other components out of service by virtue of the operation of associated protection, which disturbance can trigger chain reaction disintegrating the power network.

[005] Therefore, the most basic and integral part of all other functions including optimizations in power network operation and control is security control. Security control means controlling power flows so that no component of the network is over loaded and controlling voltages such that there is no over voltage or under voltage at any of the nodes in the network following a disturbance small or large. Security control functions or alternatively overloads alleviation and over/under voltage alleviation functions can be realized through one or combination of more controls in the network. These involve control of power flow over tie line connecting other utility network, turbine steam/water/gas input control to control real power generated by each generator, load shedding function curtails load demands of consumers, excitation controls reactive power generated by individual generator which essentially controls generator terminal voltage, transformer taps control connected node voltage, switching in/out in capacitor/reactor banks controls reactive power at the connected node. Such overload and under/over voltage alleviation functions produce changes in controlled variables/parameters in step-60 of Fig.5. In other words controlled variables/parameters are assigned or changed to the new values in step-60. This correction in controlled variables/parameters could be even optimized in case of simulation of all possible imaginable disturbances including outage of a line and loss of generation for corrective action stored and made readily available for acting upon in case the simulated disturbance actually occurs in the power network. In fact simulation of all possible imaginable disturbances is the modern practice because corrective actions need be taken before the automatic operation of individual protection of the power network components.

[006] Control of an electrical power system involving power-flow control and voltage control is performed according to the process flow diagram of fig.5. The various numbered steps in fig.5 are explained in the following.

Step-10: On-line/simulated readings of open/close status of all switches and circuit breaker are obtained, and read data of maximum and minimum reactive power generation capability limits of PV-node generators, maximum and minimum tap positions limits of tap changing transformers, and maximum power carrying capability limits of transmission lines, transformers in the power network.

Step-20: On-line readings of real and reactive power assignments or settings at PQ-nodes, real power and voltage magnitude assignments or settings at PV-nodes and transformer turns ratios are obtained. These assigned/set values are the controlled variables/parameters.

Step-30: Power flow through network components, voltage magnitude and angle at nodes, reactive power generations by generators and turns ratios of transformers in the power system are determined by performing Loadflow computation in dependence on the set of read-online/assigned/set values of controlled power network variables/parameter, in step-20 or previous process cycle step-60.

Step-40: The results of Loadflow computation of step 30 are evaluated for any over loaded network components and over/under voltage at any of nodes in the power network

Step-50: If the system state is good implying no over loaded components and no over/under voltages, the process branches to step-70, otherwise to step-60

Step-60: Changes the controlled variables/parameters set in step-20 or later set in the previous process cycle step-60 by running overload and under/over voltage alleviating function, and returns to step-30

Step-70: Actually implements assigned/set controlled variables/parameters to obtain secure/correct/acceptable operation of power system

[007] Rewritten

It is obvious that loadflow computation is performed many times in real-time operation and control environment and, therefore, **efficient and high-speed loadflow computation is necessary** to provide corrective control in the changing power system conditions including an outage or failure. Moreover, the **loadflow computation must be highly reliable to yield converged solution under wide range of system operating conditions and network parameters.** Failure to yield converged loadflow solution creates blind spot as to what

This prior art material rewritten and reorganized in the amendment copy

exactly could be happening in the network leading to potentially damaging operational and control decisions/actions in capital-intensive power utilities.

The embodiment of the present invention, the most efficient and reliable loadflow computation, as described in the above steps and in Fig.5 is very general and elaborate. The control of voltage magnitude within reactive power generation capabilities of electrical machines including generators, synchronous motors, and capacitor/inductor banks, and within operating ranges of transformer taps is normally integral part of loadflow computation as described in "LTC Transformers and MVAR violations in the Fast Decoupled Loadflow, IEEE PAS-101, No.9, PP. 3328-3332." If under/over voltage still exists in the results of loadflow computation, other control actions are taken in step-60 in the above and in Fig.5. For example, under voltage can be alleviated by shedding some of the load connected.

However, the simplest embodiment of the efficient and reliable method of loadflow computation is where only voltage magnitudes are controlled by controlling reactive power generation of generators and synchronous motors, switching in/out in capacitor/inductor banks and transformer taps. Of course, such control is possible only within reactive power capabilities of machines and capacitor/reactor banks, and within operating ranges of transformer taps. This is the case of a network in which the real power assignments have already been fixed and where steps-50 and -60 in the above and in Fig.5 are absent. Once loadflow computation is finished, the loadflow solution includes indications of reactive power generation at generator nodes and at the nodes of the capacitor/inductor banks, and indications of transformer turns ratio. Based on the known reactive power capability characteristics of the individual machines, the determined reactive power values are used to adjust the excitation current to each machine, or at least each machine presently under reactive power or VAR control, to establish the desired reactive power set points. The transformer taps are set in accordance with the turns ratio indications produced by the loadflow computations.

This procedure can be employed either on-line or off-line. In off-line operation, the user can simulate and experiment with various sets of operating conditions and determine reactive power generation and transformer tap settings requirements. A general-purpose computer can

implement the entire system. For on-line operation, the loadflow computation system is provided with data identifying the current real and reactive power assignments and transformer transformation ratios, the present status of all switches and circuit breakers in the network and machine characteristic curves in steps-10 and -20 in the above and in Fig. 5, and blocks 12, 20, 46, and 52 in Fig 6. Based on this information, a model of the system based on gain matrices of any of the invented or prior art loadflow computation methods provide the values for the corresponding node voltages, reactive power set points for each machine and the transformation ratio and tap changer position for each transformer.

The present invention relates to control of utility/industrial power networks of the types including plurality of power plants/generators and one or more motors/loads, and connected to other external utility. In the utility/industrial systems of this type it is the usual practice to adjust the real and reactive power produced by each generator and each of the other sources including synchronous condensers and capacitor/inductor banks, in order to optimize the real and reactive power generation assignments of the system. Healthy or secure operation of the network can be shifted to optimized operation through corrective control produced by optimization functions without violation of security constraints. This is referred to as security constrained optimization of operation. Such an optimization is described in the United States Patent Number: 5,081,591 dated Jan. 13, 1992: "Optimizing Reactive Power Distribution in an Industrial Power Network", where the present invention can be embodied by replacing the block nos. 56 and 66 each by a block of constant matrices [Yθ] and [YV], and replacing blocks of "Exercise Newton-Raphson Algorithm" by blocks of "Exercise Super Super Decoupled Algorithm" in places of blocks 58 and 68. **This is just to indicate the possible embodiment of the present invention in optimization functions like in many others including state estimation function. However, inventions are claimed through a simplified embodiment without optimization function as in Fig. 6 in this application.**

DISCLOSURE OF THE INVENTION

This invention relates to steady-state power network computation referred to as loadflow or Power-Flow. Loadflow computations are performed as a step in real-time operation/control and in on-line/off-line studies of Electrical Power Systems. The present invention is a parallel

algorithm for the loadflow solution. The invented parallel algorithm involving Suresh's diakoptics enabled invention of the best possible parallel computer architecture. The parallel algorithm is the best version of the many simple variants with almost similar performance.

BRIEF DESCRIPTION OF DRAWINGS

Fig. 1 is a flow-charts of the prior art methods

Fig. 2 is a one-line diagram of IEEE 14-node network and its decomposition into level-1 connectivity sub-networks

Fig. 3 is a flow-charts embodiment of the invented ~~algorithms~~

Fig. 4 is a block diagram of invented parallel computer architecture

Fig. 5 is a flow-chart of the overall controlling method for an electrical power system involving load-flow computation as a step which can be executed using one of the invented load-flow computation algorithms of Fig. 3.

Fig. 6 is a flow-chart of the simple special case of voltage control method in overall controlling method of Fig. 5 for an electrical power system

Fig. 7 is a one-line diagram of an exemplary ~~utility~~ power network having 6-nodes

SYMBOLS

The prior art and inventions will now be described using the following symbols:

$Y_{pq} = G_{pq} + jB_{pq}$	: (p-q) th element of nodal admittance matrix without shunts
$y = g_p + jb_p$	: total shunt admittance at any node-p
$V_p = e_p + jf_p = V_p\angle\theta_p$	: complex voltage of any node-p
$\overline{V}_s = e_s + jf_s = V_s\angle\theta_s$	: complex slack-node voltage
$\Delta\theta_p, \Delta V_p$	: voltage angle, magnitude corrections
$\Delta e_p, \Delta f_p$	: real, imaginary components of voltage corrections
$P_p + jQ_p$	: net nodal injected power, calculated
$\Delta P_p + j\Delta Q_p$	: nodal power residue (mismatch)
$RP_p + jRQ_p$	: modified nodal power residue
~~$PSH_p + jQSH_p$~~	~~: net nodal injected power, scheduled~~
m	: number of PQ-nodes
k	: number of PV-nodes
n=m+k+1	: total number of nodes

q>p : q is the node adjacent to node-p excluding the case of q=p

[ ] : indicates enclosed variable symbol to be a vector or a matrix

~~VSH_p : Scheduled or specified voltage magnitude at PV-node-p~~ removed

PQ-node : load-node where Real-Power-P and Reactive-Power-Q are specified

PV-node : generator-node where Real-Power-P and Voltage-Magnitude-V are specified

Bold letters represent complex quantities in the following description.

Loadflow Calculation } definition of more terms are added
~~PRIOR ART~~ Loadflow model
Loadflow method

GAUSS-SEIDEL LOADFLOW: GSL

The complex power injected into the node-p of a power network is given by the following relation,

$$P_p - jQ_p = \mathbf{V_p}^* \sum_{q=1}^{n} \mathbf{Y_{pq}V_q} = \mathbf{V_p}^* \mathbf{Y_{pp}V_p} + \mathbf{V_p}^* \sum_{q>p} \mathbf{Y_{pq}V_q} \quad (1)$$

Where,

$$P_p = \mathrm{Re}\{\mathbf{V_p}^* \sum_{q=1}^{n} \mathbf{Y_{pq}V_q}\} \quad (2)$$

$$Q_p = -\mathrm{Im}\{\mathbf{V_p}^* \sum_{q=1}^{n} \mathbf{Y_{pq}V_q}\} \quad (3)$$

Where, Re means "real part of" and Im means "imaginary part of".

The Gauss-Seidel (GS) method is used to solve a set of simultaneous algebraic equations iteratively. The GSL-method calculates complex node voltage from any node-p relation (1) as given in relation (4).

$$\mathbf{V_p} = [\{(PSH_p - jQSH_p)/\mathbf{V_p}^*\} - \sum_{q>p} \mathbf{Y_{pq}V_q}] / \mathbf{Y_{pp}} \quad (4)$$

Iteration Process

Iterations start with the experienced/reasonable/logical guess for the solution. The reference node also referred to as the slack-node voltage being specified, starting voltage guess is made for the remaining (n-1)-nodes in n-node network. Node voltage value is immediately updated with its newly calculated value in the iteration process in which one node voltage is

calculated at a time using latest updated other node voltage values. A node voltage value calculation at a time process is iterated over (n-1)-nodes in an n-node network, the reference node voltage being specified not required to be calculated.

Now, for the iteration-(r+1), the complex voltage calculation at node-p equation (4) and reactive power calculation at node-p equation (3), becomes

$$V_p^{(r+1)} = [\{(PSH_p - jQSH_p)/(V_p^*)^r\} - \sum_{q=1}^{p-1} Y_{pq} V_q^{(r+1)} - \sum_{q=p+1}^{n} Y_{pq} V_q^r] / Y_{pp} \quad (5)$$

$$Q_p^{(r+1)} = - \text{Im} \{(V_p^*)^r \sum_{q=1}^{p-1} Y_{pq} V_q^{(r+1)} - (V_p^*)^r \sum_{q=p}^{n} Y_{pq} V_q^r\} \quad (6)$$

Convergence

The iteration process is carried out until changes in the real and imaginary parts of the set of (n-1)-node voltages calculated in two consecutive iterations are all less than the specified tolerance - ε, as shown in relations (7) and (8). The lower the value of the specified tolerance for convergence check, the greater the solution accuracy.

$$|\Delta e_p^{(r+1)}| = |e_p^{(r+1)} - e_p^r| < \varepsilon \quad (7)$$

$$|\Delta f_p^{(r+1)}| = |f_p^{(r+1)} - f_p^r| < \varepsilon \quad (8)$$

Accelerated Convergence

The GS-method being inherently slow to converge, it is characterized by the use of an acceleration factor applied to the difference in calculated node voltage between two consecutive iterations to speed-up the iterative solution process. The accelerated value of node-p voltage at iteration-(r+1) is given by

$$V_p^{(r+1)} \text{(accelerated)} = V_p^r + \beta (V_p^{(r+1)} - V_p^r) \quad (9)$$

Where, β is the real number called acceleration factor, the value of which for the best possible convergence for any given network can be determined by trial solutions. The GS-method is very sensitive to the choice of β, causing very slow convergence and even divergence for the wrong choice.

Scheduled or specified voltage at a PV-node

Of the four variables, real power PSH_p and voltage magnitude VSH_p are specified at a PV-node. If the reactive power calculated using VSH_p at the PV-node is within the upper and lower generation capability limits of a PV-node generator, it is capable of holding the specified voltage at its terminal. Therefore the complex voltage calculated by relation (5) by using actually calculated reactive power Q_p in place of QSH_p is adjusted to specified voltage magnitude by relation (10).

$$V_p^{(r+1)} = (VSH_p V_p^{(r+1)})/|V_p^{(r+1)} \quad (10)$$

Computation steps for the GSL method are described in algorithm-1a, and in flow-chart of Fig. 1a.

Calculation steps of Gauss-Seidel (GSL) method

~~**Algorithm-1a: Computation steps of Gauss-Seidel Loadflow method**~~

→ Paragraph added

1. Read system data and assign an initial approximate solution. If better solution estimate is not available, set specified voltage magnitude at PV-nodes, 1.0 p.u. voltage magnitude at PQ-nodes, and all the node angles equal to that of the slack-node angle, which is referred to as the **flat-start.**
2. Form nodal admittance matrix, and Initialize iteration count r= 1
3. Scan all the node of a network, except the slack-node whose voltage having been specified need not be calculated. Initialize node count p=1, and initialize maximum change in real and imaginary parts of node voltage variables DVRMX=0.0 and DVIMX=0.0
4. Test for the type of a node at a time. For the slack-node go to step-12, for a PQ-node go to the step-9 , and for a PV-node follow the next step.
5. Compute $Q_p^{(r+1)}$ for use as an imaginary part in determining complex schedule power at a PV-node from relation (6) after adjusting its complex voltage for specified value by relation (10)
6. If $Q_p^{(r+1)}$ is greater than the upper reactive power generation capability limit of the PV-node generator, set QSH_p = the upper limit Q_p^{max} for use in relation (5), and go to step-9. If not, follow the next step.

7. If $Q_p^{(r+1)}$ is less than the lower reactive power generation capability limit of the PV-node generator, set QSH_p = the lower limit Q_p^{min} for use in relation (5), and go to step-9. If not, follow the next step.
8. Compute $V_p^{(r+1)}$ from relation (5) using $QSH_p = Q_p^{(r+1)}$, and adjust its value for specified voltage at the PV-node by relation (10), and go to step-10
9. Compute $V_p^{(r+1)}$ from relation (5)
10. Compute changes in the real and imeginary parts of the node-p voltage by using relations (7) and (8), and replace current value of DVRMX and DVIMX respectively in case any of them is larger.
11. Calculate accelerated value of $V_p^{(r+1)}$ by using relation (9), and update voltage by $V_p^r = V_p^{(r+1)}$ for immediate use in the next node voltage calculation.
12. Check for if the total numbers of nodes - n are scanned. That is if $p < n$, increment $p=p+1$, and go to step-4. Otherwise follow the next step.
13. If DVRMX and DVIMX both are not less than the convergence tolerance (ε) specified for the purpose of the accuracy of the solution, advance iteration count $r=r+1$ and go to step-3, otherwise follow the next step
14. From calculated and known values of complex voltage at different power network nodes, and tap position of tap changing transformers, calculate power flows through power network components, and reactive power generation at PV-nodes.

Decoupled Loadflow

In a class of decoupled loadflow methods, each decoupled method comprises a system of equations (11) and (12) differing in the definition of elements of [RP], [RQ], and [Yθ] and [YV]. It is a system of equations for the separate calculation of voltage angle and voltage magnitude corrections.

$$[RP] = [Y\theta]\,[\Delta\theta] \quad (11)$$

$$[RQ] = [YV]\,[\Delta V] \quad (12)$$

Successive (1θ, 1V) Iteration Scheme

In this scheme (11) and (12) are solved alternately with intermediate updating. Each iteration involves one calculation of [RP] and [Δθ] to update [θ] and then one calculation of [RQ] and [ΔV] to update [V]. The sequence of relations (13) to (16) depicts the scheme.

$$[\Delta\theta] = [Y\theta]^{-1} [RP] \quad (13)$$

$$[\theta] = [\theta] + [\Delta\theta] \quad (14)$$

$$[\Delta V] = [YV]^{-1} [RQ] \quad (15)$$

$$[V] = [V] + [\Delta V] \quad (16)$$

The scheme involves solution of system of equations (11) and (12) in an iterative manner depicted in the sequence of relations (13) to (16). This scheme requires mismatch calculation for each half-iteration; because [RP] and [RQ] are calculated always using the most recent voltage values and it is block Gauss-Seidal approach. The scheme is block successive, which imparts increased stability to the solution process. This in turn improves convergence and increases the reliability of obtaining solution.

Super Super Decoupled Loadflow: SSDL

This method is not very sensitive to the restriction applied to nodal transformation angles; SSDL **restricts transformation angles to the maximum of -48 degrees** determined experimentally for the best possible convergence from non linearity considerations, which is depicted by relations (19) and (20). However, it gives closely similar performance over wide range of restriction applied to the transformation angles say, from -36 to -90 degrees.

$$RP_p = (\Delta P_p \mathrm{Cos}\Phi_p + \Delta Q_p \mathrm{Sin}\Phi_p)/ V_p^2 \quad \text{-for PQ-nodes} \quad (17)$$

$$RQ_p = (\Delta Q_p \mathrm{Cos}\Phi_p - \Delta P_p \mathrm{Sin}\Phi_p)/V_p \quad \text{-for PQ-nodes} \quad (18)$$

$$\mathrm{Cos}\Phi_p = \text{Absolute } (B_{pp} / \sqrt{(G_{pp}^2 + B_{pp}^2)}) \geq \mathrm{Cos}(-48°) \quad (19)$$

$$\mathrm{Sin}\Phi_p = \text{-Absolute } (G_{pp} / \sqrt{(G_{pp}^2 + B_{pp}^2)}) \geq \mathrm{Sin}(-48°) \quad (20)$$

$$RP_p = \Delta P_p / (K_p V_p^2) \quad \text{-for PV-nodes} \quad (21)$$

$$K_p = \text{Absolute } (B_{pp}/Y\theta_{pp}) \quad (22)$$

$$Y\theta_{pq} = \begin{cases} -Y_{pq} & \text{-for branch r/x ratio} \le 3.0 \\ -(B_{pq} + 0.9(Y_{pq}-B_{pq})) & \text{-for branch r/x ratio} > 3.0 \\ -B_{pq} & \text{-for branches connected between two PV-nodes or a PV-node and the slack-node} \end{cases} \quad (23)$$

$$YV_{pq} = \begin{cases} -Y_{pq} & \text{-for branch r/x ratio} \le 3.0 \\ -(B_{pq} + 0.9(Y_{pq}-B_{pq})) & \text{-for branch r/x ratio} > 3.0 \end{cases} \quad (24)$$

$$Y\theta_{pp} = \sum_{q>p} -Y\theta_{pq} \quad \text{and} \quad YV_{pp} = b_p' + \sum_{q>p} -YV_{pq} \quad (25)$$

$$b_p' = (QSH_pCos\Phi_p - PSH_pSin\Phi_p / V_s^2) - b_pCos\Phi_p \quad \textbf{or}$$
$$b_p' = 2(QSH_pCos\Phi_p - PSH_pSin\Phi_p)/ V_s^2 \quad (26)$$

where, K_p is defined in relation (22), which is **initially restricted to the minimum value of 0.75 determined experimentally; however its restriction is lowered to the minimum value of 0.6 when its average over all less than 1.0 values at PV nodes is less than 0.6.** Restrictions to the factor K_p as stated in the above is system independent. However it can be tuned for the best possible convergence for any given system. **In case of systems of only PQ-nodes and without any PV-nodes, equations (23) and (24) simply be taken as $Y\theta_{pq} = YV_{pq} = -Y_{pq}$.**

Branch admittance magnitude in (23) and (24) is of the same algebraic sign as its susceptance. Elements of the two gain matrices differ in that diagonal elements of [YV] additionally contain the b' values given by relations (25) and (26) and in respect of elements corresponding to branches connected between two PV-nodes or a PV-node and the slack-node. Relations (19) and (20) with inequality sign implies that transformation angles are restricted to maximum of -48 degrees for SSDL. The method consists of relations (11) to (26). In two simple variations of the SSDL method, one is to make $YV_{pq}=Y\theta_{pq}$ and the other is to make $Y\theta_{pq}=YV_{pq}$.

SSDL method implements successive (1θ, 1V) iteration scheme and is embodied in algorithm-1b, and in flow-chart of fig.1b.

Calculation steps of super super decoupled Loadflow (SSDL) method

~~**Algorithm-1b: Computation steps of SSDL method**~~

→ a paragraph added

a. Read system data and assign an initial approximate solution. If better solution estimate is not available, set voltage magnitude and angle of all nodes equal to those of the slack-node. This is referred to as the **slack-start.**

b. Form nodal admittance matrix, and Initialize iteration count ITRP = ITRQ= r = 0

c. Compute Cosine and Sine of nodal rotation angles using relations (19) and (20), and store them. If they, respectively, are less than the Cosine and Sine of -48 degrees, equate them, respectively, to those of -48 degrees.

d. Form (m+k) x (m+k) size matrices [Yθ] and [YV] of (11) and (12) respectively each in a compact storage exploiting sparsity. The matrices are formed using relations (23), (24), (25), and (26). In [YV] matrix, replace diagonal elements corresponding to PV-nodes by very large value (say, 10.0**10). In case [YV] is of dimension (m x m), this is not required to be performed. Factorize [Yθ] and [YV] using the same ordering of nodes regardless of node-types and **store them using the same indexing and addressing information.** In case [YV] is of dimension (m x m), it is factorized using different ordering than that of [Yθ].

e. Compute residues [ΔP] at PQ- and PV-nodes and [ΔQ] at only PQ-nodes. If all are less than the tolerance (ε), proceed to step-n. Otherwise follow the next step.

f. Compute the vector of modified residues [RP] as in (17) for PQ-nodes, and using (21) and (22) for PV-nodes.

g. Solve (13) for [Δθ] and update voltage angles using, [θ] = [θ] + [Δθ].

h. Set voltage magnitudes of PV-nodes equal to the specified values, and Increment the iteration count ITRP=ITRP+1 and r=(ITRP+ITRQ)/2.

i. Compute residues [ΔP] at PQ- and PV-nodes and [ΔQ] at PQ-nodes only. If all are less than the tolerance (ε), proceed to step-n. Otherwise follow the next step.

j. Compute the vector of modified residues [RQ] as in (18) for only PQ-nodes.

k. Solve (15) for [ΔV] and update PQ-node magnitudes using [V] = [V] + [ΔV]. While solving equation (15), skip all the rows and columns corresponding to PV-nodes.

l. Calculate reactive power generation at PV-nodes and tap positions of tap changing transformers. If the maximum and minimum reactive power generation capability and

transformer tap position limits are violated, implement the violated physical limits and adjust the loadflow solution ~~by the method of reference-4~~.

m. Increment the iteration count ITRQ=ITRQ+1 and r=(ITRP+ITRQ)/2, and Proceed to step-e.

n. From calculated and known values of voltage magnitude and voltage angle at different power network nodes, and tap position of tap changing transformers, calculate power flows through power network components, and reactive power generation at PV-nodes.

SUMMARY OF THE ENVENTION
BRIEF DESCRIPTION OF DRAWINGS
DESCRIPTION OF A PREFERED EMBODYMENT
sections are added

INVENTION

Inventions include Gauss-Seidel-Patel numerical method for the solution of simultaneous algebraic equations, and parallel algorithms involving invented network decomposition technique referred to as Suresh's diakoptics and the use of the best possible parallel computer architecture invented.

The above three sections deals with ~~Patel~~ simply description of Invention

Gauss-Seidel-Patel Loadflow: GSPL

In the whole application description of claimed Invention starts from here which remains almost the same in amendment application

Gauss-seidel numerical method is well-known to be not able to converge to the high accuracy solution, which problem has been resolved for the first-time in the proposed Gauss-Seidel-Patel (GSP) numerical method.

The GSP method introduces the concept of self-iteration of each calculated variable until convergence before proceeding to calculate the next. This is achieved by replacing relation (5) by relation (27) stated in the following where self-iteration-(sr+1) over a node variable itself within the global iteration-(r+1) over (n-1) nodes in the n-node network is depicted. During the self-iteration process only V_p changes without affecting any of the terms involving V_q. At the start of the self-iteration $V_p^{sr} = V_p^{r}$, and at the convergence of the self-iteration $V_p^{(r+1)} = V_p^{(sr+1)}$.

$$(V_p^{(sr+1)})^{(r+1)} = [\{(PSH_p - jQSH_p)/((V_p^{*})^{sr})^{r}\} - \sum_{q=1}^{p-1} Y_{pq} V_q^{(r+1)} - \sum_{q=p+1}^{n} Y_{pq} V_q^{r}] / Y_{pp} \qquad (27)$$

Self-convergence

The self-iteration process is carried out until changes in the real and imaginary parts of the node-p voltage calculated in two consecutive self-iterations are less than the specified tolerance. It has been possible to establish a relationship between the tolerance specification for self-convergence and the tolerance specification for global-convergence. It is found sufficient for the self-convergence tolerance specification to be ten times the global-convergence tolerance specification.

$$|\Delta e_p^{(sr+1)}| = |e_p^{(sr+1)} - e_p^{sr}| < 10\varepsilon \quad (28)$$

$$|\Delta f_p^{(sr+1)}| = |f_p^{(sr+1)} - f_p^{sr}| < 10\varepsilon \quad (29)$$

For the global-convergence tolerance specification of 0.000001, it has been found sufficient to have the self-convergence tolerance specification of 0.00001 in order to have the maximum real and reactive power mismatches of 0.0001 in the converged solution. However, for small networks under not difficult to solve conditions they respectively could be 0.00001 and 0.0001 or 0.000001 and 0.0001, and for large networks under difficult to solve conditions they sometimes need to be respectively 0.0000001 and 0.000001.

Network Decomposition Technique: Suresh's Diakoptics

A network decomposition technique referred to as Suresh's diakoptics involves determining a sub-network for each node involving directly connected nodes referred to as level-1 nodes and their directly connected nodes referred to as level-2 nodes and so on. The level of outward connectivity for local solution of a sub-network around a given node is to be determined experimentally. This is particularly true for gain matrix based methods such as Newton-Raphson(NR), SSDL methods. Sub-networks can be solved by any of the known methods including Gauss-Seidel-Patel Loadflow (GSPL) method.

In the case of GSPL-method only one level of outward connectivity around each node is found to be sufficient for the formation of sub-networks equal to the number of nodes. Level-1 connectivity sub-networks for IEEE 14-node network is shown in Fig. 2b. The local solution of equations of each sub-network could be iterated for experimentally determined

two or more iteration. However, maximum of two iterations are fond to be sufficient. In case of GSPL-method, processing load on processors can be attempted equalization by assigning two or more smaller sub-networks to the single processor for solving separately in sequence.

Sometimes it is possible that a sub-network around any given node could be a part of the sub-network around another node making it redundant needing local solution of less than (m+k) sub-networks in case of gain matrix based methods like SSDL. Level-1 connectivity sub-networks for IEEE 14-node for parallel solution by say, SSDL-method is shown in Fig, 2b. The local solution iteration over a sub-network is not required for gain matrix based methods like SSDL. Fig. 2c shows the grouping of the non-redundant sub-networks in Fig. 2b in an attempt to equalize size of the sub-networks reducing the number of processors without increasing time for the solution of the whole network.

It should be noted that no two decomposed network parts contain the same set of nodes, or the same set of lines connected to nodes, though some same nodes could appear in two or more sub-networks.

Decomposing network in level-1 connectivity sub-networks provides for maximum possible parallelism, and hopefully fastest possible solution. However, optimum outward level of connectivity around a node sub-network can be determined experimentally for the solution of large networks by a gain matrix based method like SSDL.

Relating Sub-network Solutions to get the Network-wide Global Solution

Suresh's decomposition subroutine run by server-processor decomposes the network into sub-networks and a separate processor is assigned to solve each sub-network simultaneously in parallel. A node-p of the network could be contained in two or more sub-networks. Say a node-p is contained in or part of three sub-networks. If GSPL-method is used, voltage calculation for a node-p is performed by each of the three sub-networks. Add three voltages calculated for a node-p by three sub-networks and divide by 3 to take an average as given by relation (30).

$$V_p^{(r+1)} = (V_{p1}^{(r+1)} + V_{p2}^{(r+1)} + V_{p3}^{(r+1)})/3 \qquad (30)$$

If a gain matrix based method like SSDL is used, voltage angle correction and voltage

magnitude correction calculation for a node-p is performed by each of the three sub-networks. Add three voltage angle corrections and three voltage magnitude corrections calculated for the node-p by three sub-networks and divide by 3 to take average as given by relations (31) and (32).

$$\Delta\theta_p^{(r+1)} = (\Delta\theta_{p1}^{(r+1)} + \Delta\theta_{p2}^{(r+1)} + \Delta\theta_{p3}^{(r+1)})/3 \quad (31)$$

A Paragraph Preceding eqns. (33) and (34) are added

$$\Delta V_p^{(r+1)} = (\Delta V_{p1}^{(r+1)} + \Delta V_{p2}^{(r+1)} + \Delta V_{p3}^{(r+1)})/3 \quad (32)$$

$$e_p^{(r+1)} = (e_{p1}^{(r+1)} + e_{p2}^{(r+1)} + e_{p3}^{(r+1)} + \dots + e_{pq}^{(r+1)})/q \quad (33)$$

The relations (30), (31), and (32) can also alternatively be written as relations (34), (33), and (35).

$$t_p^{(r+1)} = (t_{p1}^{(r+1)} + t_{p2}^{(r+1)} + t_{p3}^{(r+1)} + \dots + t_{pq}^{(r+1)})/q \quad (34)$$

$$V_p^{(r+1)} = \sqrt{(\mathrm{Re}((V_{p1}^{(r+1)})^2) + \mathrm{Re}((V_{p2}^{(r+1)})^2) + \mathrm{Re}((V_{p3}^{(r+1)})^2))/3}$$

$$+ j\sqrt{(\mathrm{Im}((V_{p1}^{(r+1)})^2) + \mathrm{Im}((V_{p2}^{(r+1)})^2) + \mathrm{Im}((V_{p3}^{(r+1)})^2))/3} \quad 35$$

$$\Delta\theta_p^{(r+1)} = \sqrt{(\Delta\theta_{p1}^{(r+1)})^2 + (\Delta\theta_{p2}^{(r+1)})^2 + (\Delta\theta_{p3}^{(r+1)})^2)/3} \quad 36$$

$$\Delta V_p^{(r+1)} = \sqrt{(\Delta V_{p1}^{(r+1)})^2 + (\Delta V_{p2}^{(r+1)})^2 + (\Delta V_{p3}^{(r+1)})^2)/3} \quad 37$$

$$e_p^{(r+1)} = \sqrt{((e_{p1}^{(r+1)})^2 + (e_{p2}^{(r+1)})^2 + (e_{p3}^{(r+1)})^2 + \dots + (e_{pq}^{(r+1)})^2)/q} \quad (38)$$

(39)] added

Mathematically, square of any positive or negative number is positive. Therefore, in the above relations if the original not-squared value of any number is negative, the same algebraic sign is attached after squaring that number. Again if the mean of squared values turns out to be a negative number, negative sign is attached after taking the square root of the unsigned number.

Parallel Computer Architecture

The Suresh's diakoptics along with the technique of relating sub-network solution estimate to get the global solution estimate does not require any communication between processors assigned to solve each sub-network. All processors access the commonly shared memory through possibly separate port for each processor in a multi-port memory organization to speed-up the access. Each processor access the commonly shared memory to write results of

local solution of sub-network assigned to contribute to the generation of network-wide or global solution estimate. The generation of global solution estimate marks the end of iteration. The iteration begins by processors accessing latest global solution estimate for local solution of sub-networks assigned. That means only beginning of the solution of sub-networks by assigned processors need to be synchronized in each iteration, which synchronization can be affected by server-processor.

This leads to the invention of the simplest possible parallel computer architecture depicted in Fig, 4. In a desktop parallel computer workstation, I think it would be convenient to have input/output operation performed only by server-processor.

Distributed Computing Architecture

The parallel computer architecture depicted in Fig. 4 land itself into distributed computing architecture. This is achieved when each processor and associated memory forming a self-contained computer in itself is physically located at each network node or a substation, and communicates over communication lines with commonly shared memory and server processor both located at central station or load dispatch center in a power network. It is possible to have an input/output unit with a computer at each network node or substation, which can be used to read local sub-network data in parallel and communicate over communication line to commonly shared memory for the formation and storage of network wide global data at the central load dispatch center in the power network.

Conclusion

The inventions of Suresh's diakoptics, technique of relating local solution of sub-networks into network-wide global solution, and parallel computer architecture depicted in Fig. 4 afford an opportunity for the maximum possible parallelism with minimum possible communication and synchronization requirements. Also parallel computer architecture and parallel computer program are scalable, which is not possible with most of the parallel computers built so far. Moreover, these inventions provide bridging and unifying model for parallel computation.

Algoritham-2b: ~~Computation~~ Calculation steps for Parallel Gauss-Seidel-Patel Loadflow Method

Steps for a parallel solution of sub-networks using Gauss-Seidel-Patel method are listed as follows: added Paragraph

21. Read system data and assign an initial approximate solution. If better solution estimate is not available, set specified voltage magnitude at PV-nodes, 1.0 p.u. voltage magnitude at PQ-nodes, and all the node angles equal to that of the slack-node, which is referred to as the flat-start. The solution guess is stored in complex voltage vector say, V (I) where "I" takes values from 1 to n, the number of nodes in the whole network.

22. All processors simultaneously access network-wide global data stored in commonly shared memory, which can be under the control of server-processor, to form and locally store required admittance matrix for each sub-network.

23. Initialize complex voltage vector, say VV (I) =CMPLEX (0.0, 0.0) that receives solution contributions from sub-networks.

24. All processors simultaneously access network-wide global latest solution estimate vector V (I) available in the commonly shared memory to read into the local processor memory the required elements of the vector V (I), and perform 2-iterations of the GSPL-method in parallel for each sub-network to calculate node-voltages.

25. As soon as 2-iterations are performed for a sub-network, its new local solution estimate is contributed to the vector VV (I) in commonly shared memory under the control of server processor without any need for the synchronization. It is possible that small sub-network finished 2-iterations and already contributed to the vector VV (I) while 2-iterations are still being performed for the larger sub-network.

26. Contribution from a sub-network to the vector VV (I) means, the complex voltage estimate calculated for the nodes of the sub-network are added to the corresponding elements of the vector VV (I). After all sub-networks finished 2-iterations and contributed to the vector VV (I), its each element is divided by the number of contributions from all sub-networks to each element or divided by number of nodes directly connected to the node represented by the vector element, leading to the transformation of vector VV (I) into the new network-wide global solution estimate.

This operation is performed as indicated in relation (30) or (33). This step requires synchronization in that the division operation on each element of the vector VV(I) can be performed only after all sub-networks are solved and have made their contribution to the vector VV(I).

27. Find the maximum difference in the real and imaginary parts of [VV(I)-V(I)]

28. Calculate accelerated value of VV(I) by relation (9) as $VV(I) = V(I) + \beta [VV(I)-V(I)]$ and perform V(I)=VV(I)

29. If the maximum difference calculated in step-7 is not less than certain solution accuracy tolerance specified as stopping criteria for the iteration process, increment iteration count and go to step-3, or else follow the next step.

30. From calculated values of complex voltage at different power network nodes, and tap position of tap changing transformers, calculate power flows through power network components, and reactive power generation at PV-nodes.

It can be seen that steps 2, 4, and 5 are performed in parallel. While other steps are performed by the server-processor. However, with the refined programming, it is possible to delegate some of the server-processor tasks to the parallel-processors. For example, any assignment functions of step 1 and step 2 can be performed in parallel. Even reading of system data can be performed in parallel particularly in distributed computing environment where each sub-network data can be read in parallel by substation computers connected to operate in parallel. Flow chart for the above algorithm is given in Fig. 3b.

~~Algoritham-2b: Computation~~ Calculation steps for Parallel Super Super Decoupled Loadflow Method

Steps for a parallel solution of sub-networks using Super Super Decoupled Loadflow method are listed as follows:

41. Read system data and assign an initial approximate solution. If better solution estimate is not available, set all node voltage magnitudes and all node angles equal to those of the slack-node, which is referred to as the **slack-start**. The solution guess is stored in voltage magnitude and angle vectors say, VM (I) and VA(I) where "I" takes values from 1 to n, the number of nodes in the whole network.

42. All processors simultaneously access network-wide global data stored in commonly shared memory, which can be under the control of server-processor to form and locally store required admittance matrix for each sub-network. Form gain matrices of SSDL-method for each sub-network, factorize and store them locally in the memory associated with each processor.

43. Initialize vectors, say DVM (I)= 0.0, and DVA(I)=0.0 that receives respectively voltage magnitude corrections and voltage angle corrections contributions from sub-networks.

44. Calculate real and reactive power mismatches for all the nodes in parallel, find real power maximum mismatch and reactive power maximum mismatch by the server-computer. If both the maximum values are less then convergence tolerance specified, go to step-9. Otherwise, follow the next step.

45. All processors simultaneously access network-wide global latest solution estimate VM(I) and VA(I) available in the commonly shared memory to read into the local processor memory the required elements of the vectors VM(I) and VA(I), and perform 1-iteration of SSDL-method in parallel for each sub-network to calculate node-voltage-magnitudes and node-voltage-angles.

46. As soon as 1-iteration is performed for a sub-network, its new local solution corrections estimate are contributed to the vectors DVM (I) and DVA(I) in commonly shared memory under the control of server processor without any need for the synchronization. It is possible that small sub-network finished 1-iteration and already contributed to the vectors DVM (I) and DVA(I) while 1-iteration is still being performed for the larger sub-network.

47. Contribution from a sub-network to the vectors DVM (I) and DVA(I) means, the complex voltage estimate calculated for the nodes of the sub-network are added to the corresponding elements of the vectors DVM (I) and DVA(I). After all sub-networks finished 1-iteration and contributed to the vectors DVM (I) and DVA(I), its each element is divided by the number of contributions from all sub-networks to each element or divided by number of nodes directly connected to the node represented by the vector element, leading to the transformation of vectors DVM (I) and DVA(I) into the new network-wide global solution correction estimates. This operation is performed as indicated in relation (31) and (32) or (34) and (35). This step requires synchronization in

that the division operation on each element of the vectors DVM (I) and DVA(I) can be performed only after all sub-networks are solved and made their contribution to the vectors DVM (I) and DVA(I).

48. Update solution estimates VM(I) and VA(I), and proceed to step-3

49. From calculated values of complex voltage at different power network nodes, and tap position of tap changing transformers, calculate power flows through power network components, and reactive power generation at PV-nodes.

It can be seen that steps-2, -4, and -5 are performed in parallel. While other steps ate performed by the server-processor. However, with the refined programming, it is possible to delegate some of the server-processor tasks to the parallel-processors. For example, any assignment functions such as in step-3 can be performed in parallel. Even reading of system data can be performed in parallel particularly in distributed computing environment where each sub-network data can be read in parallel by substation computers connected to operate in parallel.

General Statements

[072] The system stores a representation of the reactive capability characteristic of each machine and these characteristics act as constraints on the reactive power, which can be calculated for each machine.

[073] While the description above refers to particular embodiments of the present invention, it will be understood that many modifications may be made without departing from the spirit thereof. The accompanying claims are intended to cover such modifications as would fall within the true scope and spirit of the present invention.

[074] The presently disclosed embodiments are therefore to be considered in all respect as illustrative and not restrictive, the scope of the invention being indicated by the appended claims in addition to the foregoing description, and all changes which come within the meaning and range of equivalency of the claims are therefore intended to be embraced therein.

REFERENCES

Foreign Patent Document

1. US Patent Number: 4868410 dated September 19, 1989: "System of Load Flow Calculation for Electric Power System"
2. US Patent Number: 5081591 dated January 14, 1992: "Optimizing Reactive Power Distribution in an Industrial Power Network"

Published Pending Patent Applications

31. Canadian Patent Application Number: CA2107388 dated 9 November, 1993: "System of Fast Super Decoupled Loadflow Computation for Electrical Power System"
42. International Patent Application Number: PCT/CA/2003/001312 dated 29 August, 2003: "System of Super Super Decoupled Loadflow Computation for Electrical Power System"

Other Publications

51. Stagg G.W. and El-Abiad A.H., "Computer methods in Power System Analysis", McGrow-Hill, New York, 1968
62. S.B.Patel, "Fast Super Decoupled Loadflow", IEE proceedings Part-C, Vol.139, No.1, pp. 13-20, January 1992
73. Shin-Der Chen, Jiann-Fuh Chen, "Fast loadflow using multiprocessors", Electrical Power & Energy Systems, 22 (2000) 231-236

CLAIMS

The present invention is applicable to systems to process load flow computation by means of parallel algorithm using invented (Fig.4)/available parallel computer. The embodiments of the invention in which an exclusive property or privilege is claimed are defined as follows:

1. A method of controlling voltages and power flows in a power network, comprising the steps of:

   obtaining on-line data of open/close status of switches and circuit breakers in the power network,

   obtaining on-line readings of real and reactive power assignments or settings at PQ-nodes, real power and voltage magnitude assignments or settings at PV-nodes and transformer turns ratios, which are the controlled variables/parameters,

   performing load-flow computation to calculate complex voltages or voltage magnitude corrections and voltage angle corrections at the power network nodes providing for the calculation of power flowing through different network components, and reactive power generation and transformer tap-position indications,

   evaluating the computed load flow for any of the over loaded power network components and for under/over voltage at any of the nodes,

   correcting one or more controlled parameters and repeating the computing and evaluating steps until evaluating step finds a good power system without any over loaded components and without any under/over voltages in the power network, and

   effecting a change in the power flowing through network components and voltages and phases at the nodes of the power network by actually implementing the finally obtained values of controlled parameters after evaluating step finds a good power system.

2. A loadflow computation defined in claim-1 is characterized in using self-iteration over a node voltage calculation within a network-wide global iteration as depicted by the following iteration-(r+1) equation elaborated for clarity

$$(V_p^{(sr+1)})^{(r+1)} = [\{(PSH_p - jQSH_p)/((V_p^*)^{sr})^r\} - \sum_{q=1}^{p-1} Y_{pq} V_q^{(r+1)} - \sum_{q=p+1}^{n} Y_{pq} V_q^r] / Y_{pp} \quad (27)$$

3. A load-flow computation as defined in claim1 is characterized in the use of a method of decomposing a network referred to as Suresh's diakoptics that involves determining a sub-network for each node involving directly connected nodes referred to as level-1 nodes and their directly connected nodes referred to as level-2 nodes and so on, and the level of outward connectivity for local solution of a sub-network around a given node is to be determined experimentally.

4. A load-flow computation as defined in claim-1 is characterized in the use of Parallel solving of all sub-networks defined in claim-3 using available solution estimate at the start of the iteration, initializing a vector of dimension equal to the number of nodes with each element value zero, adding solution estimates for a node resulting from different sub-networks in a corresponding vector element, counting the number of additions and calculating new solution estimate or corrections to the available solution estimate using any relevant relations in the following with the number '3' replaced by the actual number of contributions or additions to a vector element, and storing it as initial available estimate for the next iteration

Claims 3 & 4 are made Part of claim 1

$$V_p^{(r+1)} = (V_{p1}^{(r+1)} + V_{p2}^{(r+1)} + V_{p3}^{(r+1)})/3 \quad (30)$$

$$\Delta\theta_p^{(r+1)} = (\Delta\theta_{p1}^{(r+1)} + \Delta\theta_{p2}^{(r+1)} + \Delta\theta_{p3}^{(r+1)})/3 \quad (31)$$

$$\Delta V_p^{(r+1)} = (\Delta V_{p1}^{(r+1)} + \Delta V_{p2}^{(r+1)} + \Delta V_{p3}^{(r+1)})/3 \quad (32)$$

(33) (34) added

$$V_p^{(r+1)} = \sqrt{(Re((V_{p1}^{(r+1)})^2) + Re((V_{p2}^{(r+1)})^2) + Re((V_{p3}^{(r+1)})^2))/3} + j\sqrt{(Im((V_{p1}^{(r+1)})^2) + Im((V_{p2}^{(r+1)})^2) + Im((V_{p3}^{(r+1)})^2))/3} \quad (33)$$

$$\Delta\theta_p^{(r+1)} = \sqrt{(\Delta\theta_{p1}^{(r+1)})^2 + (\Delta\theta_{p2}^{(r+1)})^2 + (\Delta\theta_{p3}^{(r+1)})^2)/3} \quad (34)$$

$$\Delta V_p^{(r+1)} = \sqrt{(\Delta V_{p1}^{(r+1)})^2 + (\Delta V_{p2}^{(r+1)})^2 + (\Delta V_{p3}^{(r+1)})^2)/3} \quad (35)$$

(36) (37) (38) (39) added

35. A load-flow computation as defined in claim-1. Claim-2, claim-3 and claim-4 is characterized in the use of the simplified parallel computer a server processor-array processors architecture, where each of the array processors communicate only with server processor and commonly shared memory locations and not among themselves. (Fig.4)

4.
5.
6.
7.
8.
9.

Claims 4-9 are added in the same manner of the claims in reference-2 US Patent no. 5081591 issued in 1992

~~This app~~

The amendment of this application follows the style of description of reference-2 US Patent no. 5081591 issued in 1992.

Abstract is rewritten to meet the requirement as brought by the examiner Mr. Victor J. Taylor in his report dated April 2, 2008

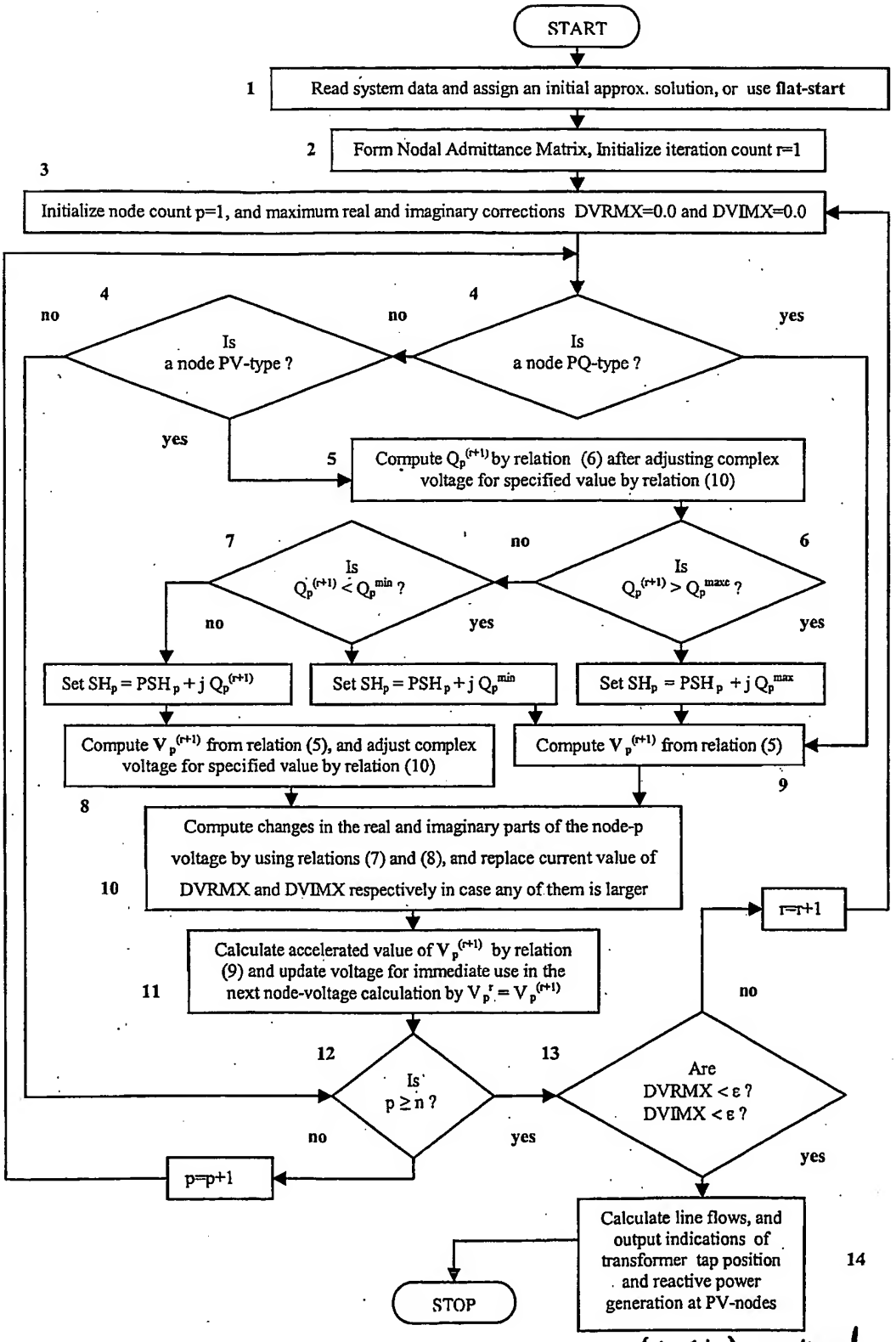


Fig. 1a: Prior Art: flow-chart of Gauss-Seidel Loadflow ~~Algorithm-1a~~ (GSL) method

~~1/9~~ 1/10

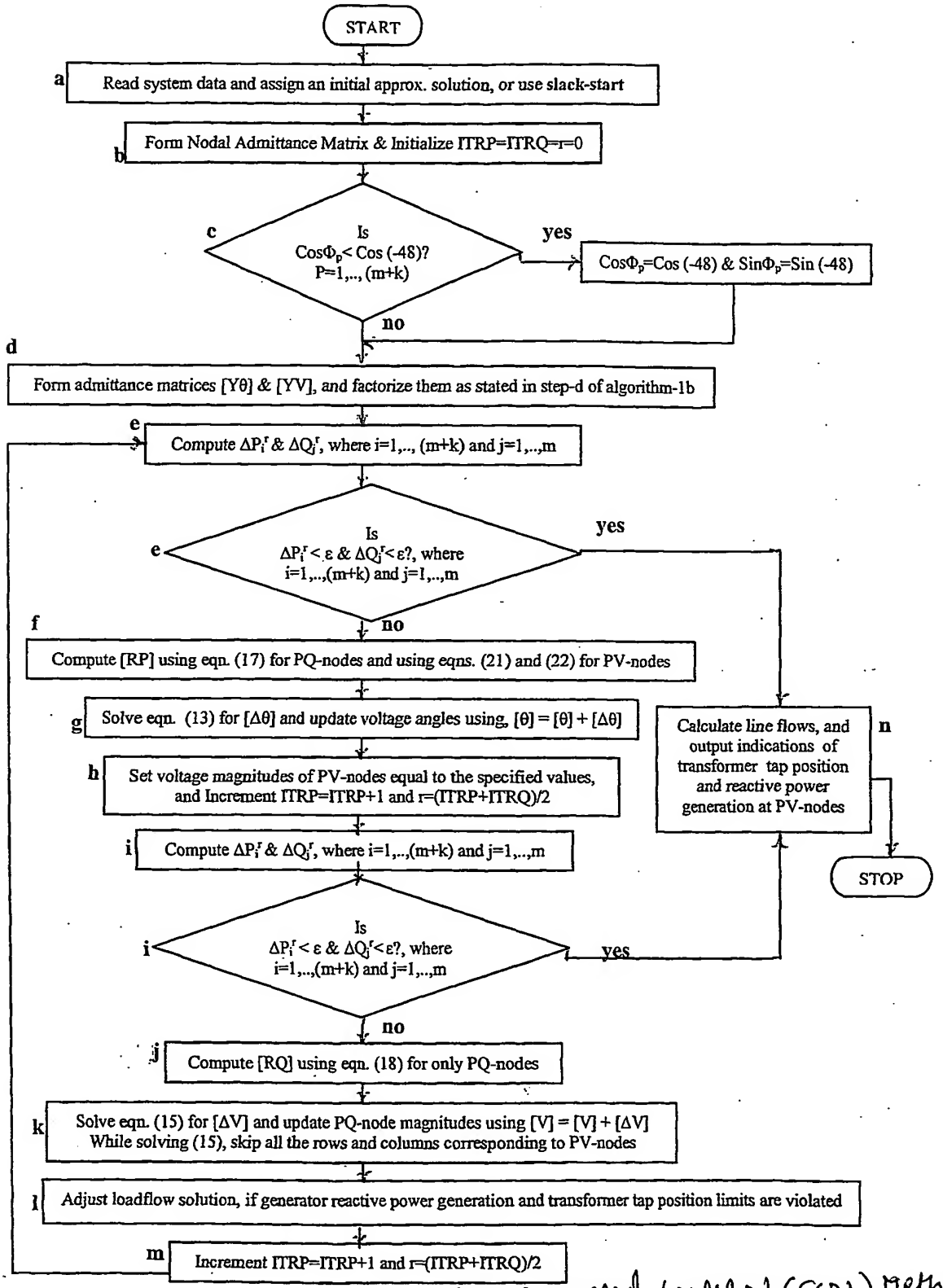


Fig.1b: Prior art: Flow-chart ~~of SSDL solution algorithm-1b~~ : Super Super Decoupled Loadflow (SSDL) Method

2/9/10

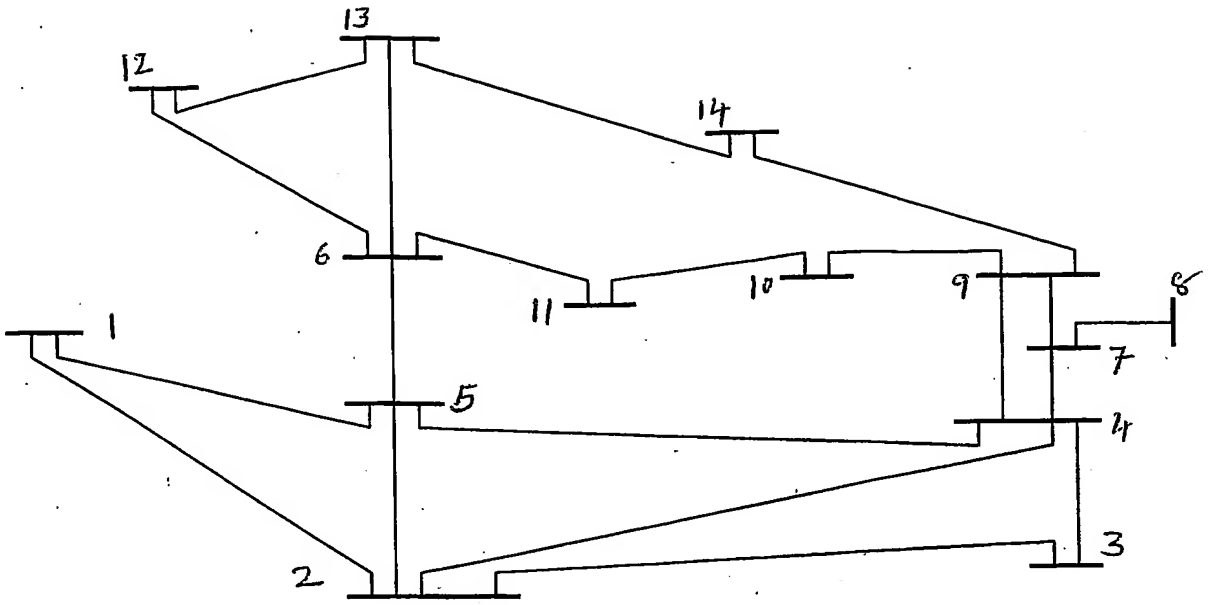


Fig. 2a: One-line diagram of IEEE 14-node network

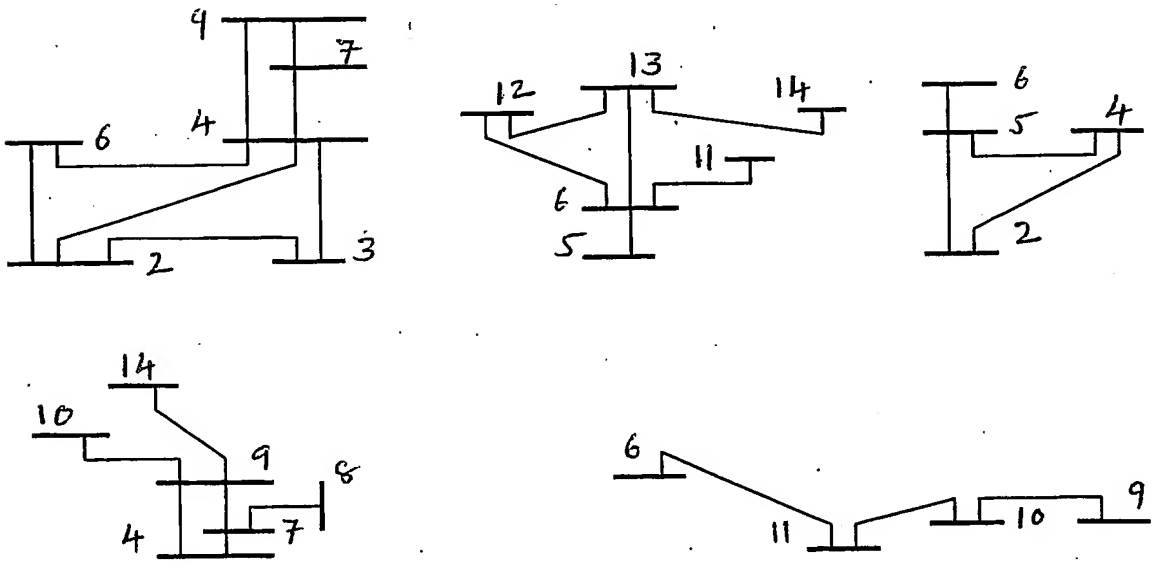


Fig. 2c: Non-redundant Level-1 sub-networks of fig. 2b are regrouped to reduce the number of processors required without increasing the number of nodes in any regrouped sub-network larger than the original largest sub-network of 6-nodes

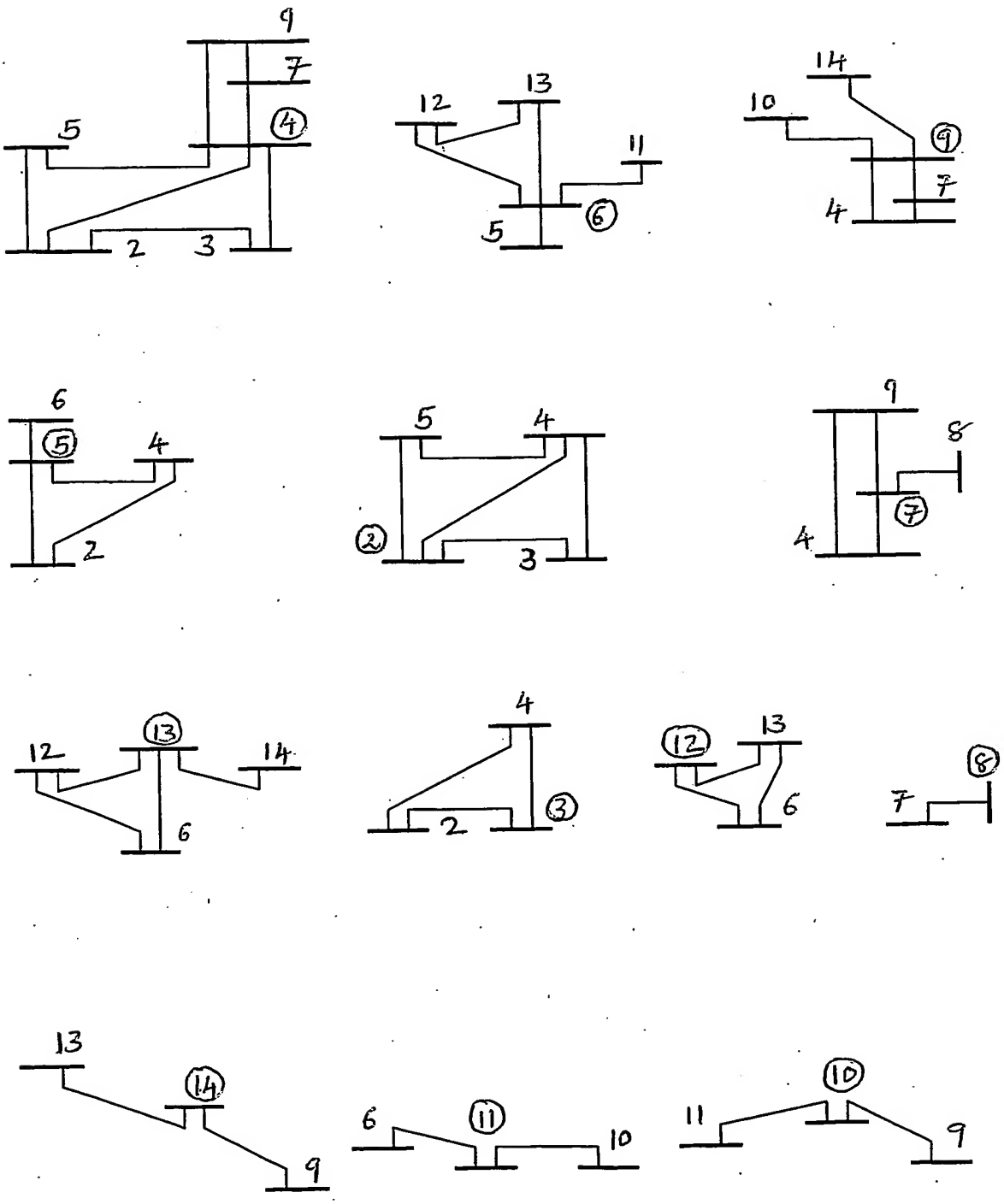


Fig. 2b: Level-1 sub-networks around circled nodes for the network of fig. 2a

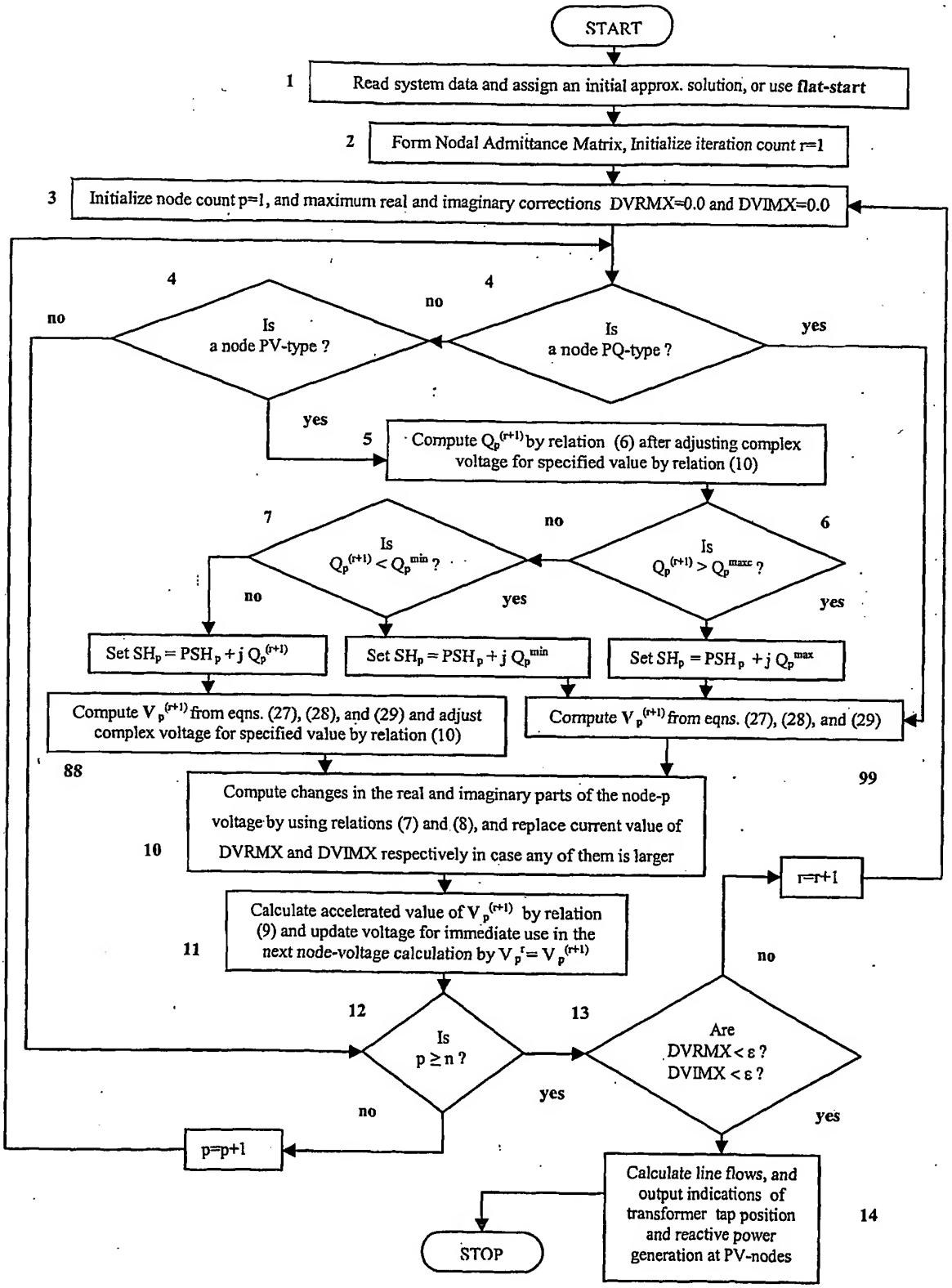


Fig. 3a: Invention: flow-chart of Gauss-Seidel-Patel Loadflow Algorithm-2a

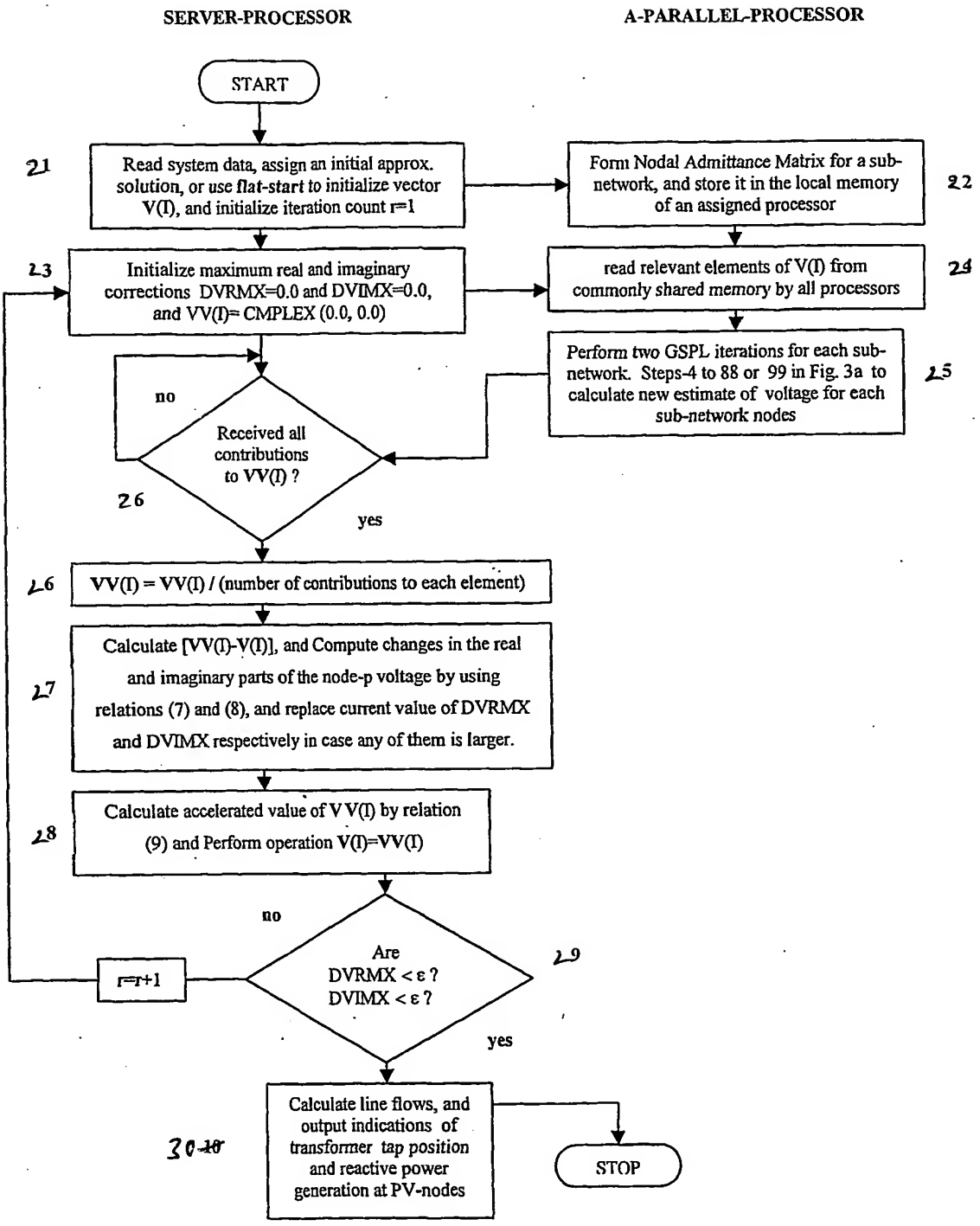


Fig. 3b: **Invention: flow-chart of Parallel-Gauss-Seidel-Patel Loadflow ~~Algorithm-2b~~ (PGSPL) Method**

6/9/10

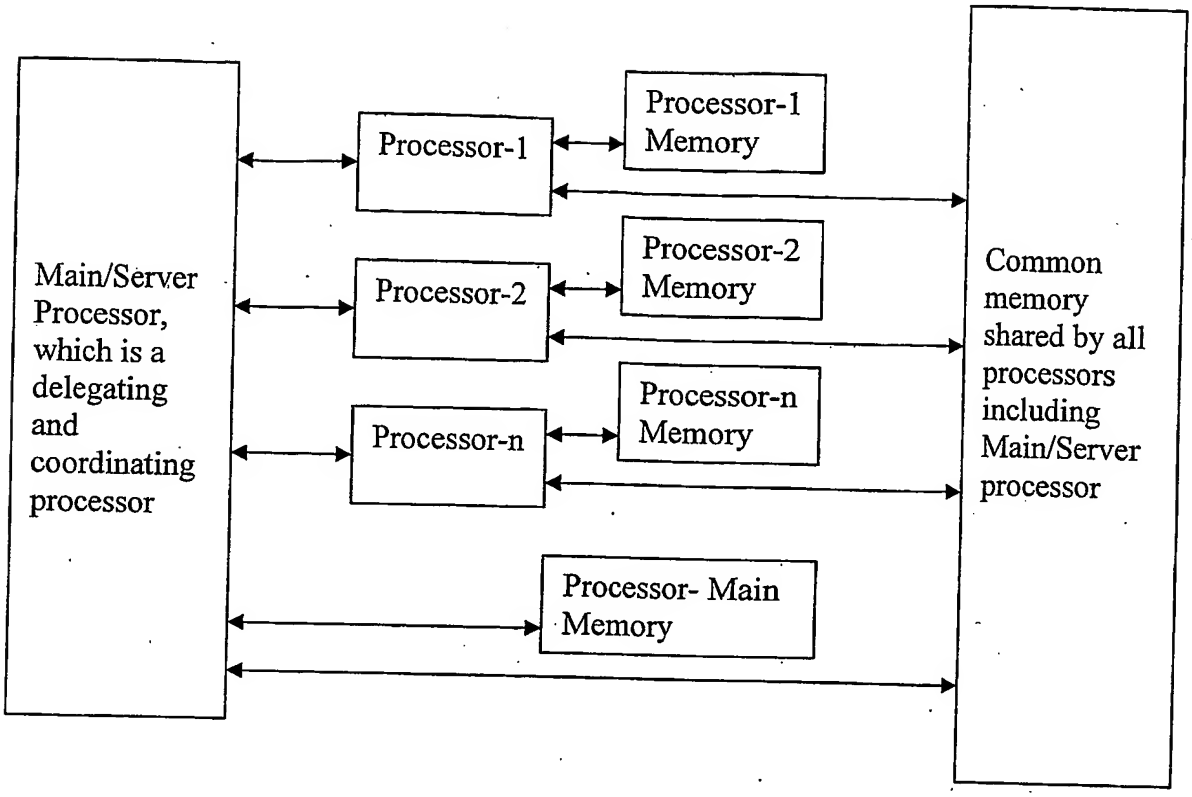
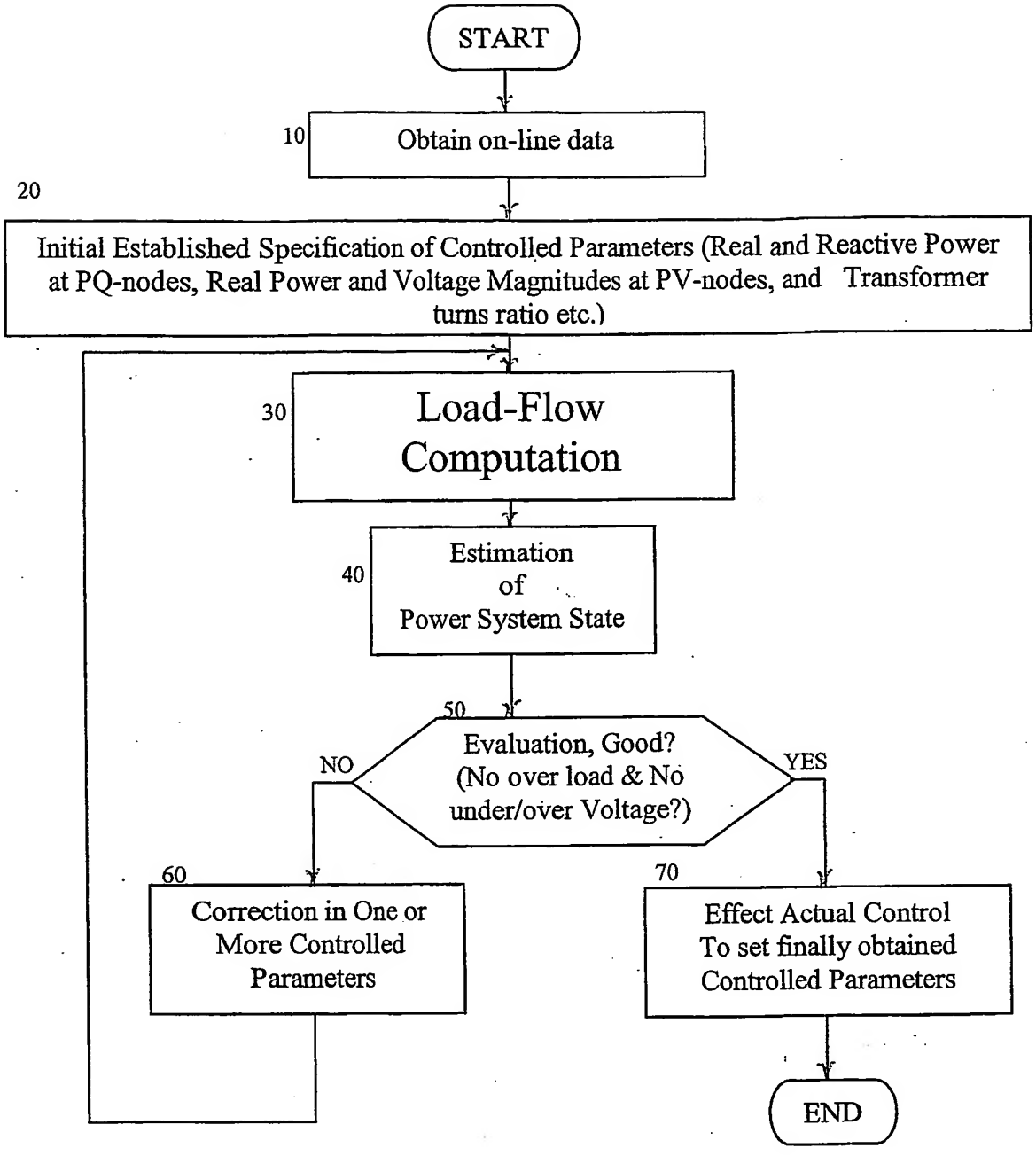


Fig. 4: Invented Parallel computer Architecture/organization



Loadflow Calculation in Powerflow control ~~and~~/and/or

Fig. 5: ~~Load-Flow Computation in Security Control of Electrical Power System (PRIOR ART)~~

Voltage control in Electrical Power system

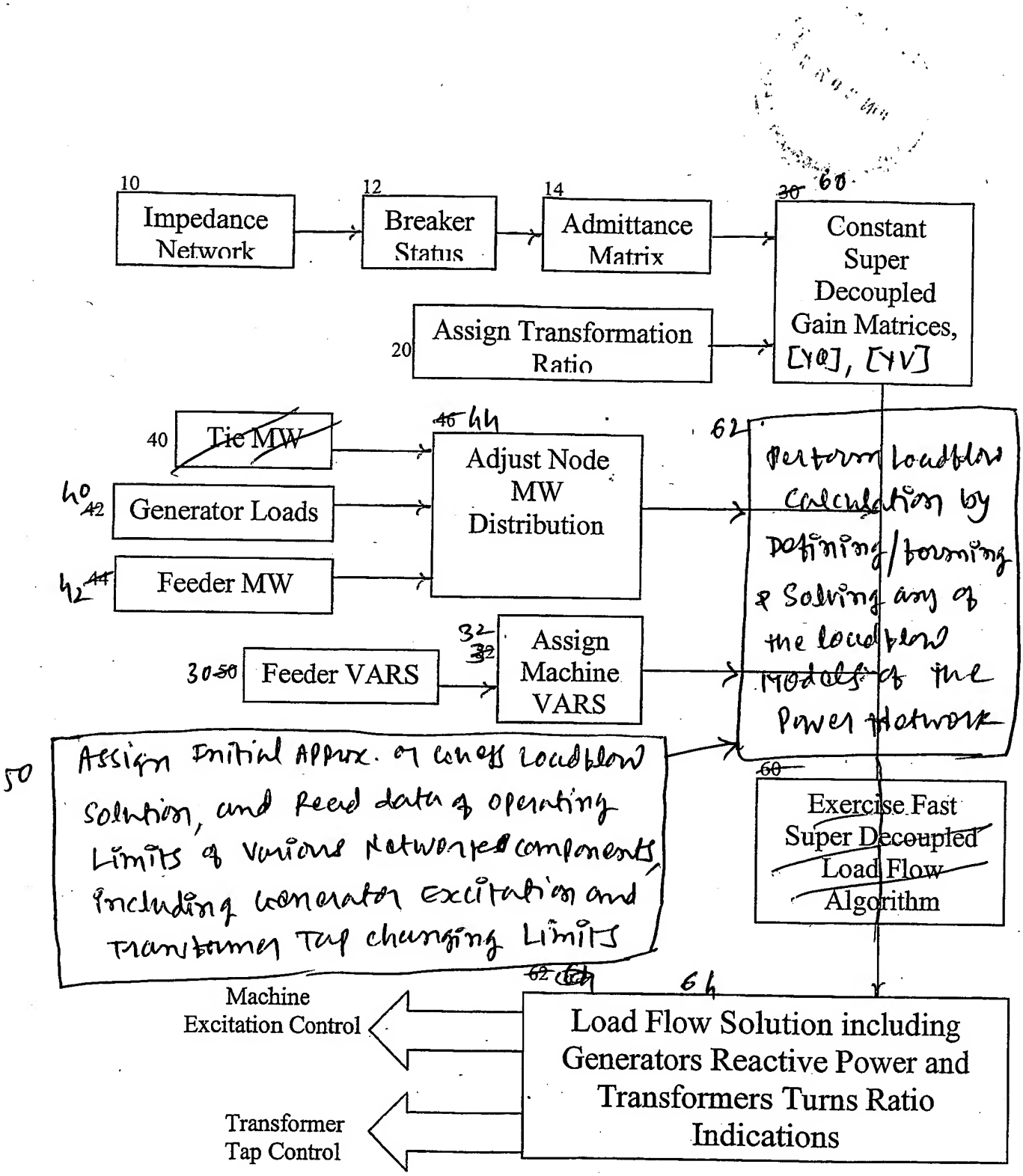


Fig. 6: Load-Flow Computation ~~in~~ for Voltage Control ~~of~~ in Electrical Power System ~~(PRIOR ART)~~

9/10

Fig. 7 An exemplary 6-node power system is added on separate page - 10/10